B

EDLIN
The Line Editor

- Introduction
- Command Information
- How To Start EDLIN
- Using EDLIN Commands
- Special Editing Keys
- Command Options
- Error Messages

Introduction

This appendix is a function and command reference to the EDLIN program on your System diskette. EDLIN is provided for use by persons with prior experience in programming computers. EDLIN is not intended for novice users. If you are new to using computers, you will find that a dedicated word processing program may be easier to use. The information in this appendix does not include a tutorial for use by beginners.

If you are an experienced computer user you may choose to use EDLIN to create, change, and display files, whether they are source program or text files.

You can use EDLIN to:

- Create new source files and save them
- Update existing files and save both the updated and original files
- Delete, edit, insert, and display lines
- Search for, delete, or replace text within one or more lines.

The text in files created or edited by EDLIN is divided into lines, each up to 253 characters long. Line numbers are generated and displayed by EDLIN during the editing process, but are not actually present in the saved file.

When you insert lines, all line numbers following the inserted text advance automatically by the number of lines being inserted.

When you delete lines in a file, all line numbers following the deleted text decrease automatically by the number of lines deleted. As a result, lines are always numbered consecutively in your file.

EDLIN

EDLIN commands perform editing functions on lines of text. The following list contains information you should read before you use EDLIN commands.

1 Pathnames are acceptable as options to commands.

For example, typing EDLIN \BIN\USER\JOE \TEXT.TXT allows you to edit the TEXT.TXT file in the subdirectory \JOE.

2 You can reference line numbers relative to the current line (the line with the asterisk). Use a minus sign with a number to indicate lines before the current line. Use a plus sign with a number to indicate lines after the current line.

Example:

```
-10,+10L
```

This command lists 10 lines before the current line, the current line, and 10 lines after the current line.

3 Multiple commands may be issued on one command line. When you issue a command to edit a single line using a line number (<line>), a semicolon must separate commands on the line. Otherwise, one command may follow another without any special separators. In the case of a Search or Replace command, the <string> may be ended by a **CTRL Z** instead of a **RETURN**.

Examples:

The following command line edits line 15 and then displays lines 10 through 20 on the screen.

```
15;-5, +5L
```

The command line in the next example searches for "This string" and then displays 5 lines before and 5 lines after the line containing the matched string. If the search fails, then the displayed lines are those line numbers relative to the current line.

```
SThis string CTRL Z-5, +5L
```

4 You can type EDLIN commands with or without a space between the line number and command.

For example, to delete line 6, the command 6D is the same as 6 D.

5 It is possible to insert a control character (such as CONTROL-C) into text by using the quote character CONTROL-V before the capital letter associated with it while in insert mode. CONTROL-V tells MS-DOS to recognize the next capital letter typed as a control

character. It is also possible to use a control character in any of the string arguments of Search or Replace by using the special quote character. For example:

S **CTRL V** **Z**

finds the first occurrence of CONTROL-Z in a file

R **CTRL V** Z **CTRL Z** foo

replaces all occurrences of CONTROL-Z in a file by foo

S **CTRL V C** CTRL Z bar

replaces all occurrences of CONTROL-C by bar

It is possible to insert CONTROL-V into the text by typing CONTROL-V-V.

6 The CONTROL-Z character ordinarily tells EDLIN, "This is the end of the file." If you have CONTROL-Z characters elsewhere in your file, you must tell EDLIN that these other control characters do not mean end-of-file. Use the /B option to tell EDLIN to ignore any CONTROL-Z characters in the file and to show you the entire file.

EDLIN Commands

The EDLIN commands are summarized in the following table. They are also described in further detail following the description of command options.

Command	Purpose
<line>	Edits line no.
A	Appends lines
C	Copies lines
D	Deletes lines
E	Ends editing
I	Inserts lines
L	Lists text
M	Moves lines
P	Pages text
Q	Quits editing
R	Replaces lines
S	Searches text
T	Transfers text
W	Writes lines

How To Start EDLIN

To start EDLIN, type:

```
EDLIN <filespec>
```

If you are creating a new file, the <filespec> should be the name of the file you wish to create. If EDLIN does not find this file on a drive, EDLIN creates a new file with the name you specify. The following message and prompt are displayed:

```
New file
*_
```

Notice that the prompt for EDLIN is an asterisk (*).

You can now type lines of text into your new file. To begin entering text, you must enter an I (Insert) command to insert lines. The I command is discussed later in this chapter.

If you want to edit an existing file, <filespec> should be the name of the file you want to edit. When EDLIN finds the file you specify on the designated or default drive, the file is loaded into memory.

If the entire file can be loaded, EDLIN displays the following message on your screen:

```
End of input file
*
```

You can then edit the file using EDLIN editing commands. If the file is too large to be loaded into memory, EDLIN loads lines until memory is 3/4 full, then displays the * prompt. You can then edit the portion of the file that is in memory.

To edit the remainder of the file, you must save some of the edited lines on disk to free memory; then EDLIN can load the unedited lines from disk into memory. Refer to the Write and Append commands in this Appendix for more information.

When you complete the editing session, you can save the original and the updated (new) files by using the End command. The End command is discussed in this appendix in the section **EDLIN Commands**. The original file is renamed with an extension of .BAK, and the new file has the filename and extension you specify in the EDLIN command.

The original .BAK file is not erased until the end of the editing session, or until disk space is needed by the editor (EDLIN).

Do not try to edit a file with a filename extension of .BAK because EDLIN assumes that any .BAK file is a backup file.

If you find it necessary to edit such a file, rename the file with another extension (using the MS-DOS RENAME command discussed in Chapter 5), then start EDLIN and specify the new <filespec>.

Special Editing Keys

The table below describes some of the commands, codes, and functions that are assigned to the special editing keys.

The "command line" is the line on your screen where you enter the text and EDLIN commands for the line you are editing. A "template" is the line you create, in part, from the line you are editing.

Special Editing Keys

Function	Key	Description
Copy 1 character	**F1**	Copies 1 character from the template to the new line.
Copy up to character	**F2**	Copies all characters from the template to the new line, up to the character specified.
Copy template	**F3**	Copies all remaining characters in the template to the screen.
Skip one character	**DEL**	Does not copy (skips over) a character.

Skip up to character	**F4**	Does not copy (skips over) the characters in the template, up to the character specified.
Quit input	**ESC**	Voids the current input; leaves the template unchanged.
Insert mode	**INS**	Enters/exits insert mode.
Replace mode	**INS F3**	Turns insert mode off; this is the default.
New template	**F5**	Makes the new line the new template.

The EDLIN Keys

Key **F1**

Purpose Copies one character from the template to the command line.

Comments Pressing the **F1** key copies one character from the template to the command line. When the **F1** key is pressed, one character is inserted in the command line and the insert key is disabled.

Example Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Pressing the **F1** key copies the first character (T) to the second of the two lines displayed:

```
   1:*This is a sample file
F1 1:*T_
```

Each time the **F1** key is pressed, one more character appears:

```
F1 1:*Th_
F1 1:*Thi_
F1 1:*This_
```

Key **F2**

Purpose Copies multiple characters up to a given character.

Comments Pressing the **F2** key copies all characters up to a given character from the template to the command line. The given character is the next character typed after **F2**; it is not copied or displayed on the screen.

Pressing the **F2** key causes the cursor to move to the single character that is specified in the command. If the template does not contain the specified character, nothing is copied. Pressing **F2** also disables the insert key.

Example Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line.

Pressing the **F2** key copies all characters up to the character specified immediately after the **F2** key.

```
    1:*This is a sample file
F2p 1:*This is a sam_
```

Key **F3**

Purpose Copies template to command line.

Comments Pressing the **F3** key copies all remaining characters from the template to the command line. Regardless of the cursor position at the time the **F3** key is pressed, the rest of the line appears, and the cursor is positioned after the last character on the line.

The insert key is disabled.

Example Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line.

Pressing the **F3** key copies all characters from the template cursor.

```
   1:*This is a sample file (template)
F3 1:*This is a sample file._ (command line)
```

Key **DEL**

Purpose Skips over one character in the template.

Comments Pressing the **DEL** key skips over one character in the template. Each time you press the **DEL** key, one character is not copied from the template. The action of the **DEL** key is similar to the **F1** key, except that **DEL** skips a character in the template rather than copying it to the command line.

Example Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Pressing the **DEL** key skips over the first character (T).

```
    1:*This is a sample file
DEL 1:*_
```

The cursor position does not change and only the template is affected. To see how much of the line has been skipped over, press the **F3** key, to move the cursor beyond the last character of the line.

```
    1:*This is a sample file.
DEL 1:*_
F3  1:*his is a sample file._
```

Key **F4**

Purpose Skips multiple characters in the template up to the specified character.

Comments Pressing the **F4** key skips over all characters up to a given character in the template. This character is not copied and is not shown on the screen. If the template does not contain the specified character, nothing is skipped over.

The action of the **F4** key is similar to the **F2** key, except that **F4** skips over characters in the template rather than copying them to the command line.

Example Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Pressing the **F4** key skips over all the characters in the template up to the character pressed after the **F4** key:

```
    1:*This is a sample file
F4p 1:*_
```

The cursor position does not change. To see how much of the line has been skipped over, press the **F3** key to copy the template. This moves the cursor beyond the last character of the line:

```
      1:*This is a sample file:
F4pF3 1:*ple file._
```

Key **ESC**

Purpose Quits input and empties the command line.

Comments Pressing the **ESC** key empties the command line, but it leaves the template unchanged. **ESC** also prints a backslash (\), carriage return, and line feed, and turns insert mode off. The cursor (indicated by the underline) is positioned at the beginning of the line. Pressing the **F3** key copies the template to the command line and the command line is identical to the original template.

Example Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Assume that you want to replace the line with "Sample File:"

```
1:*This is a sample file.
1:*Sample File:_
```

To cancel the line you just entered (Sample File), and to keep “This is a sample file.”, press **ESC**. Notice that a backslash appears on the Sample File line to tell you it has been cancelled.

```
    1:*This is a sample file.
ESC 1:*Sample File    \
    1:_
```

Press **RETURN** to keep the original line, or to perform any other editing functions. If **F3** is pressed, the original template is copied to the command line:

```
F3 1: This is a sample file._
```

Key **INS**

Purpose Enters/exits insert mode.

Comments Pressing the **INS** key causes EDLIN to enter or exit insert mode. The current cursor position in the template is not changed. The cursor moves as each character is inserted. However, when you have finished inserting characters, the cursor is positioned at the same character as it was before the insertion began. Thus, characters are inserted **in front of** the character that the cursor points to.

Example Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Assume that you press the **F2** and **f** keys:

```
    1:*This is a sample file
F2f 1:*This is a sample _
```

Now press the **INS** key and insert the characters "edit" and a space:

```
    1:*This is a sample file.
F2f 1:*This is a sample _
INSedit 1:*This is a sample edit _
```

If you now press the **F3** key, the rest of the template is copied to the line:

```
    1:*This is a sample edit _
F3  1:*This is a sample edit file._
```

If you pressed the **RETURN** key, the remainder of the template would be truncated, and the command line would end at the end of the insert:

```
INSedit RETURN 1:*This is a sample edit _
```

To exit insert mode, simply press the **INS** key again.

Key **INS F3**

Purpose Enters replace mode.

Comments Pressing the **INS** and **F3** keys simultaneously causes EDLIN to exit insert mode and enter replace mode.

All the characters you type replace characters in the template. When you start to edit a line, replace mode is in effect.

If the **RETURN** key is pressed, the remainder of the template is deleted.

Example Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line.

Assume that you then press **F2**m, **INSlary, INS F3 tax, and then F3:**

```
1:*This is a sample file.
F2m 1:*This is a sa_
INSlary 1:*This is a salary_
INS F3 tax 1:*This is a salary tax_
F3 1:*This is a salary tax file._
```

Notice that you inserted **lary** and replaced **mple** with tax. If characters on the command line extend beyond the length of the template, the remaining characters in the template are automatically appended when you press **F3**.

Key **F5**

Purpose Creates a new template.

Comments Pressing the **F5** key copies the current command line to the template. The contents of the old template are deleted. Pressing **F5** outputs an @ ("at sign" character), a carriage return, and a line feed. The command line is also emptied and insert mode is turned off.

NOTE **F5** performs the same function as the **ESC** key, except that the template is changed and an @ ("at sign" character) is printed instead of a \.

Example Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line.

Assume that you enter **F2**m, **INS**lary, **INS F3** tax, and then **F3**:

```
1:*This is a sample file.
F2m 1:*This is a sa_
INSlary 1:*This is a salary_
INS F3 tax 1:*This is a salary tax_
F3 1:*This is a salary tax file._
```

At this point, assume that you want this line to be the new template, so you press the **F5** key:

```
F5 1:*This is a salary tax file.@
```

The @ indicates that this new line is now the new template. Additional editing can be done using the new template.

Command Options

Several EDLIN commands accept one or more options. The effect of a command option varies, depending on with which command it is used. The following list describes each option.

<line>

<line> indicates a line number that you type. Line numbers must be separated by a comma or a space from other line numbers, other options, and from the command.

<line> may be specified one of three ways:

Number

Any number less than 65534. If a number larger than the largest existing line number is specified, then <line> means the line after the last line number.

Period (.)

If a period is specified for <line>, then <line> means the current line number. The current line is the last line edited, and is not necessarily the last line displayed. The current line is marked on your screen by an asterisk (*) between the line number and the first character.

Pound (#).

The pound sign indicates the line after the last line number. If you specify # for <line>, this has the same effect as specifying a number larger than the last line number.

RETURN A carriage return entered without any of the <line> specifiers listed above directs EDLIN to use a default value appropriate to the command.

? The question mark option directs EDLIN to ask you if the correct string has been found. The question mark is used only with the Replace and Search commands. Before continuing, EDLIN waits for either a **Y** or **RETURN** for a yes response, or for any other key for a no response.

<**string**> <string> represents text to be found, to be replaced, or to replace other text. The <string> option is used only with the Search and Replace commands. Each <string> must be ended by a **CTRL Z** or a **RETURN** (see the Replace command for details). No spaces should be left between strings or between a string and its command letter, unless you want those spaces to be part of the string.

(A)ppend

Purpose Adds the specified number of lines from disk to the file being edited in memory. The lines are added at the end of lines that are currently in memory.

Syntax [<**n**>**A**]

Comments This command is meaningful only if the file being edited is too large to fit into memory. As many lines as possible are read into memory for editing when you start EDLIN.

To edit the remainder of the file that does not fit into memory, lines that have already been edited must be written to disk. Then you can load unedited lines from disk into memory with the Append command.

Refer to the Write command in this appendix for information on how to write edited lines to disk.

NOTES

1. If you do not specify the number of lines to append, lines are appended to memory until available memory is 3/4 full. No action is taken if available memory is already 3/4 full.

2. The message "End of input file" is displayed when the Append command has read the last line of the file into memory.

(C)opy

Purpose Copies a range of lines to a specified line number. The lines can be copied as many times as you want by using the <count> option.

Syntax [<**line**>],[<**line**>],<**line**>[,<**count**>]**C**

Comments If you do not specify a number in <count>, EDLIN copies the lines one time. If the first or the second <line> is omitted, the default is the current line. The file is renumbered automatically after the copy.

The line numbers must not overlap or an "Entry error" message appears. For example, **3,20,15C** would result in an error message.

Examples Assume that the following file exists and is ready to edit:

```
1: This is a sample file
2: used to show copying lines.
3: See what happens when you use
4: the Copy command
5: (the C command)
6: to copy text in your file.
```

You can copy this entire block of text by issuing the following command:

```
1,6,7C
```

The result is:

```
1: This is a sample file
2: used to show copying lines.
3: See what happens when you use
4: the Copy command
5: (the C command)
6: to copy text in your file.
7: This is a sample file
8: used to show copying lines.
9: See what happens when you use
10: the Copy command
11: (the C command)
12: to copy text in your file.
```

If you want to place the text within other text, the third <line> option should specify the line before which you want the **copied** text to appear. For example, assume that you want to copy lines and **insert** them within the following file:

```
1: This is a sample file
2: used to show copying lines.
3: See what happens when you use
4: the Copy command
5: (the C command)
```

```
 6: to copy text in your file.
 7: You can also use COPY
 8: to copy lines of text
 9: to the middle of your file.
10: End of sample file.
```

The command **3,6,10C** results in the following file:

```
 1: This is a sample file
 2: used to show copying lines.
 3: See what happens when you use
 4: the Copy command
 5: (the C command)
 6: to copy text in your file.
 7: You can also use COPY
 8: to copy lines of text
 9: to the middle of your file.
10: See what happens when you use
11: the Copy command
12: (the C command)
13: to copy text in your file.
14: End of sample file.
```

(D)elete

Purpose

Deletes a specified range of lines in a file.

Syntax

[<**line**>][**,**<**line**>]**D**

Comments

If the first <line> is omitted, that option defaults to the current line (the line with the asterisk next to the line number). If the second <line> is omitted, then just the first <line> is deleted. When lines have been deleted, the line immediately after the deleted section becomes the current line and has the same line number as the first deleted <line> had before the deletion occurred.

Examples

Assume that the following file exists and is ready to edit:

```
 1: This is a sample file
 2: used to show dynamic line numbers.
 3: See what happens when you use
 4: Delete and Insert
    .
    .
    .
25: (the D and I commands)
26: to edit the text
27:*in your file.
```

To delete multiple lines, type

<line>,<line>D:
5,24D

The result is:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit text
7:*in your file.
```

To delete a single line, type

6D

The result is:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6:*in your file.
```

Next, delete a range of lines from the following file:

```
1: This is a sample file
2: used to show dynamic line numbers.
3:*See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit text
7: in your file.
```

To delete a range of lines beginning with the current line, type:

```
,6D
```

The result is:

```
1: This is a sample file
2: used to show dynamic line numbers.
3:*in your file.
```

Notice that the lines are automatically renumbered.

<line> Edit

Purpose Edits line of text.

Syntax [<**line**>]

Comments When a line number is typed, EDLIN displays the line number and text; then, on the line below, EDLIN reprints the line number. The line is now ready for editing. You may use any of the EDLIN editing commands to edit the line. The existing text of the line serves as the template until the **RETURN** key is pressed.

WARNING If the **RETURN** key is pressed while the cursor is in the middle of the line, the remainder of the line is deleted.

Example Assume that the following file exists and is ready to edit:

```
1: This is a sample file.
2: used to show
3: the editing of line
4:*four.
```

To edit line 4, type:

```
4
```

The contents of the line are displayed with a cursor below the line:

```
4:* four.
4:*_
```

Now, using the **F3** special editing key, type:

```
INSnumber 4: number_
F3 RETURN 4: number four.
5:*_
```

(E)nd

Purpose Ends the editing session.

Syntax **E**

Comments This command saves the edited file on disk, renames the original input file <filename>.BAK, and then exits EDLIN. If the file was created during the editing session, no .BAK file is created.

The E command takes no options. When you begin EDLIN, you must enter the file specification for the output file. If the drive is not selected when EDLIN is started, the file is saved on the disk in the default drive. It is still possible to COPY the file to a different drive using the MS-DOS COPY command.

You must be sure that the disk contains enough free space for the entire file. If the disk does not contain enough free space, the write is aborted and the edited file lost, although part of the file might be written out to the disk.

Example **E RETURN**

After execution of the E command, the MS-DOS default drive prompt (for example, A>) is displayed.

(I)nsert

Purpose Inserts text immediately before the specified <line>.

Syntax **[<line>]I**

Comments If you are creating a new file, the I command must be given before text can be typed (inserted). Text begins with line number 1. Successive line numbers appear automatically each time **RETURN** is pressed.

EDLIN remains in insert mode until **CTRL Z** is typed.

When the insert is completed and insert mode is finished, the line immediately following the inserted lines becomes the current line. All line numbers following the inserted section are incremented by the number of lines inserted.

If <line> is not specified, the default is the current line number and the lines are inserted immediately before the current line.

If <line> is any number larger than the last line number, or if a pound sign (#) is specified as <line>, the inserted lines are appended to the end of the file. In this case, the last line inserted becomes the current line.

Examples

Assume that the following file exists and is ready to edit:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit text
7:*in your file.
```

To insert text before a specific line that is **not** the current line, type <line>I:

```
7I
```

The result is:

```
7:_
```

Now, type the new text for line 7:

```
7: and renumber lines
```

Then to end the insertion, press **CTRL Z** on the **next** line:

```
8: CTRL    Z
```

Now type L to list the file. The result is:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit text
7: and renumber lines
8:*in your file.
```

To insert lines immediately before the current line, type:

I

The result is:

```
8: _
```

Now, insert the following text and terminate with a **CTRL Z** on the next line:

```
8: so they are consecutive
9: CTRL Z
```

Now to list the file and see the result, type

L

The result is:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit text
7: and renumber lines
8: so they are consecutive
9:*in your file.
```

To append new lines to the end of the file, type:

10I

This produces the following:

```
10: _
```

Now, type the following new lines:

```
10: The insert command can place new lines
11: in the file; there's no problem
12: because the line numbers are dynamic;
13: they'll go all the way to 65533.
```

End the insertion by pressing **CTRL Z** on line 14. The new lines appear at the end of all previous lines in the file. Now type the List command, L:

The result is:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
6: to edit text
7: and renumber lines
8: so they are consecutive
9: in your file.
10: The insert command can place new lines
11: in the file; there’s no problem
12: because the line numbers are dynamic;
13: they’ll go all the way to 65533.
```

(L)ist

Purpose Lists a range of lines, including the two lines specified.

Syntax [<**line**>][,<**line**>]**L**

Comments Default values are provided if either one or both of the options are omitted. If you omit the first option, as in:

```
,<line>L
```

the display starts 11 lines before the current line and ends with the specified <line>. The beginning comma is required to indicate the omitted first option.

NOTE If the specified <line> is more than 11 lines before the current line, the display is the same as if you omitted both options. If you omit the second option, as in

```
<line>L
```

23 lines are displayed, starting with the specified <line>.

If you omit both parameters, as in

```
L
```

23 lines are displayed — the 11 lines before the current line, the current line, and the 11 lines after the current line. If there are less than 11 lines before the current line, more than 11 lines after the current line are displayed to make a total of 23 lines.

Examples Assume that the following file exists and is ready to edit:

```
1: This is a sample file
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
.
.
.
15:*The current line contains an asterisk.
.
.
26: to edit text
27: in your file.
```

To list a range of lines without reference to the current line, type <line>,<line>L:

2,5L

The result is:

```
2: used to show dynamic line numbers.
3: See what happens when you use
4: Delete and Insert
5: (the D and I commands)
```

To list a range of lines beginning with the current line, type ,<line> L:

,26L

The result is:

```
15:*The current line contains an asterisk.
.
.
.
26: to edit text
```

To list a range of 23 lines centered around the current line, type only L:

```
L
```

The result is:

```
4: Delete and Insert
5: (the D and I commands)
    .
    .
    .
3: The current line is listed in the middle of the
   range.
4: The current line remains unchanged by the L
   command.
5:*The current line contains an asterisk.
    .
    .
    .
6: to edit text.
```

(M)ove

Purpose Moves a range of text to the line specified.

Syntax [<**line**>],[<**line**>],<**line**>**M**

Comments Use the Move command to move a block of text (from the first <line> to the second <line>) to another location in the file.

Text is inserted before the target line.

The lines are renumbered according to the direction of the move. For example,

```
,+25,100M
```

moves the text from the current line plus 25 lines to line 100. If the line numbers overlap, EDLIN displays an “Entry error” message.

(P)age

Purpose Pages through a file 23 lines at a time.

Syntax [<**line**>][**,**<**line**>]**P**

Comments If the first <line> is omitted, that number defaults to the current line plus one. If the second <line> is omitted, 23 lines are listed. The new current line becomes the last line displayed and is marked with an asterisk.

(Q)uit

Purpose Quits the editing session, does not save **any** editing changes, and exits to the MS-DOS operating system.

Syntax **Q**

Comments EDLIN prompts you to make sure you don't want to save the changes.

Type **Y** if you want to quit the editing session.

No editing changes are saved and no .BAK file is created. Refer to the End command in this Appendix for information about the .BAK file.

Type **N** or any other character except Y if you want to continue the editing session.

NOTE When started, EDLIN erases any previous copy of the file with an extension of .BAK to make room to save the new copy. If you reply **Y** to the Abort edit (Y/N)? message, your previous backup copy no longer exists.

Example

```
Q
Abort edit (Y/N)?Y      RETURN
A>_
```

(R)eplace

Purpose Replaces all occurrences of a string of text in the specified range with a different string of text or blanks.

Syntax **[<line>][,<line>][?]R<string1> CTRL Z <string2>**

Comments As each occurrence of <string1> is found, it is replaced by <string2>. Each line in which a replacement occurs is displayed. If a line contains two or more replacements of <string1> with <string2>, then the line is displayed once for each occurrence.

When all occurrences of <string1> in the specified range are replaced by <string2>, the R command terminates and the asterisk prompt reappears.

If a second string is to be given as a replacement, then <string1> must be separated from <string2> with a **CTRL Z**. <String2> must end with a **CTRL Z RETURN** combination or with a simple **RETURN**.

If <string1> is omitted, then Replace takes the old <string1> as its value. If there is no old <string1>, i.e., this is the first replace done, then the replacement process ends immediately.

If the first <line> is omitted in the range argument (as in ,<line>) then the first <line> defaults to the line **after** the current line. If the second <line> is omitted (as in <line> or <line>,), the second <line> defaults to #. Therefore, R<string1> <string2> is the same as <line> + 1,#. Remember that # indicates the line after the last line of text.

If <string1> is ended with a **CTRL Z** and there is no <string2>, <string2> is taken as an empty string and becomes the new replace string. For example,

```
R<string1> CTRL Z RETURN
```

deletes all of <string1>, but

```
R<string1>   RETURN
```

and

```
R RETURN
```

both use the old <string2> and the later example also uses the old <string1>. Note that "old" here refers to a previous string specified either in a Search or a Replace command.

If the question mark (?) option is used, the Replace command stops at each line with a string that matches <string1>, displays the line with <string2> in place, and then displays the prompt O.K.?. If you press **Y** or the **RETURN** key, then <string2> replaces <string1>, and the next occurrence of <string1> is found.

Again, the O.K.? prompt is displayed. This process continues until the end of the range or until the end of the file. After the last occurrence of <string1> is found, EDLIN displays the asterisk prompt.

If you press any key besides **Y** or **RETURN** after the O.K.? prompt, <string1> is left as it was in the line, and Replace goes to the next occurrence of <string1>. If <string1> occurs more than once in a line, each occurrence of <string1> is replaced individually, and the O.K.? prompt is displayed after each replacement. In this way, only the desired <string1> is replaced, and you can prevent unwanted substitutions.

Examples

Assume that the following file exists and is ready for editing:

```
 1: This is a sample file
 2: used to show dynamic line numbers.
 3: See what happens when you use
 4: Delete and Insert
 5: (the D and I commands)
 6: to edit text
 7: in your file.
 8: The insert command can place new lines
 9: in the file; there's no problem
10: because the line numbers are dynamic;
11: they'll go all the way to 65533.
   *
```

To replace all occurrences of <string1> with <string2> in a specified range, type:

```
2,12 Rand CTRL Z or RETURN
```

The result is:

```
4: Delete or Insert
5: (the D or I commors)
8: The insert commors can place new lines
```

Note that in the above replacement, some unwanted substitutions have occurred. To avoid these and to confirm each replacement, the same original file can be used with a slightly different command.

In the next example, to replace only certain occurrences of the first <string> with the second <string>, type:

2? Rand CTRL Z or RETURN

The result is:

```
4: Delete or Insert
   O.K.? Y
5: (The D or I commands)
   O.K.? Y
5: (The D or I commors)
   O.K.? N
8: The insert commor can place new lines
   O.K.? N
*_
```

Now, type the List command **(L)** to see the result of all these changes:

```
.
.
4: Delete or Insert
5: (The D or I commands)
.
8: The insert command can place new lines
.
```

(S)earch

Purpose

Searches the specified range of lines for a specified string of text.

Syntax

[<line>][,<line>][?]S<string> RETURN

Comments

The <string> must be ended with a **RETURN**. The first line that matches <string> is displayed and becomes the current line. If the question mark option is not specified, the Search command ends when a match is found. If no line contains a match for <string>, the message "Not found" is displayed.

If the question mark option (?) is included in the command, EDLIN displays the first line with a matching string; it then prompts you with the message O.K.?. If you press either the **Y** or **RETURN** key, the line becomes the current line and the search ends.

If you press any other key, the search continues until another match is found, or until all lines have been searched (and the Not found message is displayed). If the first <line> is omitted (as in first <line> defaults to the line **after** the current line. If the second <line> is omitted (as in <line> S<string> or <line>, S<string>), the second <line> defaults to # (line after last line of file).

This is the same as typing <line>, #S <string>.

If <string> is omitted, Search takes the old string if there is one. (Note that "old" here refers to a string specified in a previous Search or Replace command.) If there is not an old string (i.e., no previous search or replace has been done), the command ends immediately.

Examples

Assume that the following file exists and is ready for editing:

```
 1: This is a sample file
 2: used to show dynamic line numbers.
 3: See what happens when you use
 4: Delete and Insert
 5: (the D and I commands)
 6: to edit text
 7: in your file.
 8: The insert command can place new lines
 9: in the file; there's no problem
10: because the line numbers are dynamic;
11:*they'll go all the way to 65533.
```

To search for the first occurrence of the string "and", type

1,12 Sand RETURN

The following line is displayed:

```
4: Delete and Insert
```

To get the "and" in line 5, modify the search command by typing:

DEL   F3 ,12 Sand RETURN

The search then continues from the line after the current line (line 4), since no first line was given. The result is:

```
5: (the D and I commands)
```

To search through several occurrences of a string until the correct string is found, type:

1, ? Sand

The result is:

```
4: Delete and Insert
   O.K.?_
```

If you press any key (except **Y** or **RETURN**), the search continues, so type N here:

```
O.K.? N
```

Continue:

```
5: (the D and I commands)
   O.K.?_
```

Now press **Y** to terminate the search:

```
O.K.? Y
*_
```

To search for string XYZ from the line after the current line to the end without the verification (O.K.?), type:

```
SXYZ
```

EDLIN reports a match and continues to search for the same string when you issue the S command:

```
S
```

Transfer

Purpose Inserts (merges) the contents of <filespec> into the file currently being edited at <line>. If <line> is omitted, then the current line is used.

Syntax [**<line>**]**T**<**filespec**>

Comments This command is useful if you want to put the contents of a file into another file or into the text you are typing. The transferred text is inserted at the line number specified by <line> and the lines are renumbered.

(W)rite

Purpose Writes a specified number of lines to disk from the lines that are being edited in memory. Lines are written to disk beginning with line number 1.

Syntax [<**n**>]**W**

Comments This command is meaningful only if the file you are editing is too large to fit into memory. When you start EDLIN, EDLIN reads lines into memory until memory is 3/4 full.

To edit the remainder of your file, you must write edited lines in memory to disk. Then you can load additional unedited lines from disk into memory by using the Append command.

NOTE If you do not specify the number of lines, lines are written until memory is 3/4 full. No action is taken if available memory is already less than 3/4 full. All lines are renumbered, so that the first remaining line becomes line number 1.

Error Messages

When EDLIN finds an error, one of the following error messages is displayed:

Cannot edit .BAK file—rename file

You attempted to edit a file with a filename extension of .BAK. .BAK files cannot be edited because this extension is reserved for backup copies.

If you need the .BAK file for editing purposes, you must either RENAME the file with a different extension; or COPY the .BAK file and give it a different filename extension.

No room in directory for file

When you attempted to create a new file, either the file directory was full or you specified an illegal disk drive or an illegal filename.

Check the command line that started EDLIN for illegal filename and illegal disk drive entries. If the command is no longer on the screen and if you have not yet typed a new command, the EDLIN start command can be recovered by pressing the **F3** key. If this command line contains no illegal entries, run the CHKDSK program for the specified disk drive. If the status report shows that the disk directory is full, remove the disk. Insert and format a new disk.

Entry Error

The last command typed contained a syntax error.

Retype the command with the correct syntax and press **RETURN**.

Line too long

During a Replace command, the string given as the replacement caused the line to expand beyond the limit of 253 characters. EDLIN aborted the Replace command.

Divide the long line into two lines, then try the Replace command.

Disk Full—file write not completed

You gave the End command, but the disk did not contain enough free space for the whole file. EDLIN aborted the E command and returned you to the operating system. Some of the file may have been written to the disk.

Only a portion (if any) of the file has been saved. You should probably delete that portion of the file and restart the editing session. The file is not available after this error. Always be sure that the disk has sufficient free space for the file to be written to disk **before** you begin your editing session.

Incorrect DOS version

You attempted to run EDLIN under a version of MS-DOS that was not 2.0 or higher.

You must make sure that the version of MS-DOS that you are using is 2.0 or higher.

Invalid drive name or file

You have not specified a valid drive or filename when starting EDLIN.

Specify the correct drive or filename.

Filename must be specified

You did not specify a filename when you started EDLIN.

Specify a filename.

Insufficient memory

There is not enough memory to run EDLIN.

You must free some memory by saving the contents of memory to disk or by clearing memory before restarting EDLIN.

File not found

The filename specified during a Transfer command was not found.

Specify a valid filename when issuing a Transfer command.

Must specify destination number

A destination line number was not specified for a Copy or Move command.

Reissue the command with a destination line number.

Not enough room to merge the entire file

There was not enough room in memory to hold the file during a Transfer command.

You must free some memory by writing some files to disk or by deleting some files before you can transfer this file.

File creation error

The EDLIN temporary file cannot be created.

Check to make sure that the directory has enough space to create the temporary file. Also, make sure that the file does not have the same name as a subdirectory in the directory where the file to be edited is located.